Assignment 7: BST Sort

Overview

You will utilize a Binary Search Tree (BST) to sort a fairly large set of data, and write a nice visualization of the unsorted and sorted result. BSTs can be used for sorting in the following manner:

Let D=(d_1, d_2, .. , d_n) be the ordered set of n unsorted data elements (note: there is a difference between the words "ordered" and "sorted" here. Please think about it to refine your scientific communication skills). Then:

- insert the elements subsequently into a BST
- traverse the BST depth first in inorder.

The output of that algorithm is the sorted data set.

About tree traversal: we will talk about traversal order in class on monday. You can also prepare yourself and read about it in the textbook. There are 3 different modes of traversing a tree depth first:

- preorder
- inorder
- postorder

They yield different results. Only inorder traversal guarantees a sorted output. However, in this assignment, you need to program all 3 modes (i.e., one method that can traverse the tree in any of the three modes), to see and experience the difference.

You will visualize the data using SimpleGUI, and it will look like this (Figure 1):

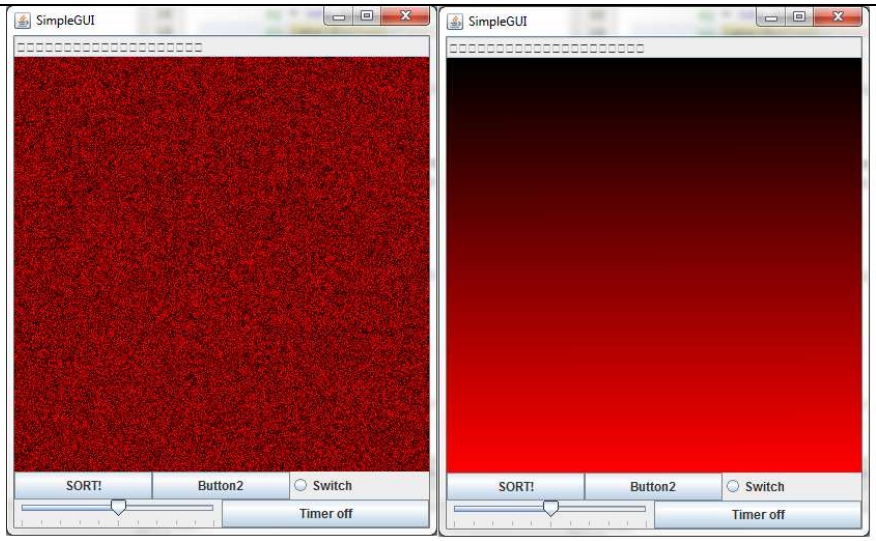

Figure 1. Left: unsorted data set. Right: sorted, using inorder BST traversal.

Your Task in Steps:

You need to create your own BST data structure. However, only insertion and traversal is needed. These are the simple methods when handling trees. I spare you the torture of node deletion in a tree. Your tree should hold data of type "double". We use the data as the key, i.e. there is no additional data assigned to the node.

1. create a BST data structure. This needs a class "BSTNode", and a class "BST".
2. in BST, implement a method "public boolean insert(double d)", which inserts the value "d" into the tree **if it is not already in the tree**. If successfully inserted, return true, otherwise ("d" is already present in the tree) return false.
3. in BST, implement a method "LinkedList<Double> traverse(String mode)", which takes a String input argument (either "inorder", "preorder", or "postorder"), and creates a LinkedList<Double> by traversing the BST accordingly.
4. In your Main class (third class of this assignment), do the following:
    a. declare an int variable SIZE = 200. This will determine the number of data elements, and the output window size.
    b. create a LinkedList<Double>, and add SIZE*SIZE random values to it. We will assume here, that the random generator does create SIZE*SIZE unique values.
    c. create a method "visualizeList" and visualize this list (see below)
    d. insert all values into your BST
    e. traverse the tree in mode "inorder", which yields a second list.
    f. visualize the second list.
    g. In between d and e, wait for a button-click.
5. How to visualize the data: the lists holds double values, which were created between [0..1). You can plot this sequence of numbers in a square of size SIZE x SIZE with dots of a color determined by the data value.
    a. create a SimpleGUI of size (SIZE, SIZE). Iterate through the list (for-each loop), incrementing an index variable, which tells you which element (index) you just accessed. You compute the x and y coordinates to plot by:
        i. x = index % SIZE
        ii. y = index / SIZE
    b. You compute the color to plot the data by:
        i. Color c = new Color(value*256, 0, 0)

You will see, that inorder traversal gives you a nice, smooth red gradient. Please experiment with the other traversals (pre/post order). What happens if you utilize these to sort? The answer with this random data set is somewhat surprising, you will see!

Good luck.

Points: BST data structure: 5 points. Visualization: 3 points. Framework (Main class etc): 2 points.